RouteLens: Easy Route Following for Map Applications

Jessalyn Alvina, Caroline Appert, Olivier Chapuis, Emmanuel Pietriga

RouteLens: Easy Route Following for Map Applications

Jessalyn Alvina[1,2] Caroline Appert[1,2] Olivier Chapuis[1,2] Emmanuel Pietriga[3,2]

[1]Univ Paris-Sud & CNRS (LRI)
F-91405 Orsay, France

[2]Inria
F-91405 Orsay, France

[3]Inria Chile (CIRIC)
CL-7561211 Santiago, Chile

ABSTRACT

Millions of people go to the Web to search for geographical itineraries. Inspecting those map itineraries remains tedious because they seldom fit on screen, requiring much panning & zooming to see details. Focus+context techniques address this problem by displaying routes at a scale that allows them to fully fit on screen: users see the entire route at once, and perform magnified steering using a lens to navigate along the path, revealing additional detail. Navigation based on magnified steering has been shown to outperform pan & zoom for large steering tasks. Yet, this task remains challenging, in part because paths have a tendency to "slip off" the side of the lens. RouteLenses automatically adjust their position based on the geometry of the path that users steer through. RouteLenses make it easier for users to follow a route, yet do not constrain movements too strictly, leaving them free to move the lens away from the path to explore its surroundings.

Categories and Subject Descriptors

H.5.2 [**[Information Interfaces and Presentation]**]: User Interfaces - Graphical user interfaces.

General Terms

Design, Human Factors, Experimentation, Performance

Keywords

Focus+Context, Route, Map, Steering law.

1. INTRODUCTION

Web-based mapping service applications have become the tool of choice for exploring geographical areas and locating points of interest. Millions of people, both novice and expert computer users, go on sites such as Google Maps, OpenStreetMap or Bing Maps to search for itineraries, compare them, and eventually select one. This task remains tedious because map itineraries seldom fit on screen, requiring much panning & zooming of the map to see particular details. For instance, a tourist faced with several alternative itineraries to go from one place to another in a city may want to explore each one at street level to pick the one that passes through the most places of interest. To address this problem, Web-based mapping services feature panning & zooming capabilities, augmented with an overview widget to prevent excessive context loss when zooming in. However, the scale difference between the overview and the detailed view remains limited. Furthermore, decoupling the overview from the detailed view is not necessarily the best-suited strategy to effectively support navigation along itineraries.

Focus+context techniques offer an alternative, well-suited to following lengthy routes at varying levels of magnification. The route is displayed at a scale that allows it to fully fit on screen, and the focus+context technique, typically a fisheye lens [14, 13], provides an in-place magnification of the locally-bounded region of interest around the cursor. Users see the entire route at once, and perform *magnified steering* [8] to navigate along the path, displaying – in-context and in more detail – the portion that falls below the lens. Navigation based on magnified steering has been shown to outperform pan & zoom for large steering tasks [8]. Yet, this task remains a challenging one for users, in part because paths have a tendency to "slip off" the side of the lens.

We introduce RouteLens, a new content-aware technique that automatically adjusts the lens' position based on the geometry of the path that users steer through, so as to keep the lens on track in case of overshoot (Figure 1). RouteLens makes it easier for users to follow a route, yet do not constrain movements too strictly. The lens is more or less strongly attracted to the path depending on its distance to it, and users remain free to move the lens away from it to explore its surroundings. RouteLenses only affect the motor behavior of lenses, and can easily be combined with any type of graphical

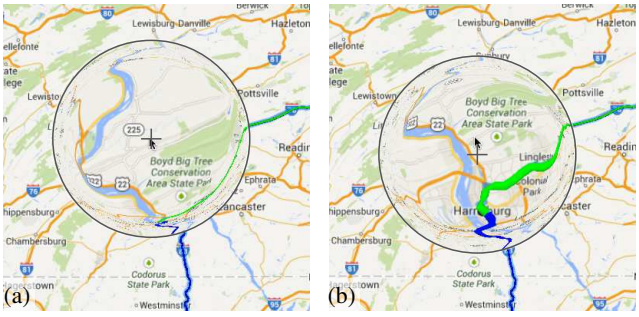

Figure 1: Following an itinerary. (a) Conventional lens: the user overshoots at a right turn in Harrisburg; losing the route that falls in the distorted region. (b) RouteLens: the route's attraction compensates the overshoot; the lens remains closer to the route, which remains in focus.

Jessalyn Alvina, Caroline Appert, Olivier Chapuis & Emmanuel Pietriga. RouteLens: Easy Route Following for Map Applications. In AVI '14: Proceedings of the International Working Conference on Advanced Visual Interfaces, 125-128, ACM, may 2014.

http://dx.doi.org/10.1145/2598153.2598200

magnification lens [4, 13]. After a brief overview of related work, we describe the behavior and implementation of RouteLenses. We then report on a laboratory experiment in which RouteLenses improve user performance over conventional magnification lenses on a path following task, as predicted by steering law [1].

2. RELATED WORK

There are two main ways of showing itineraries in mapping applications: representing them as a list of turn-by-turn driving instructions, or as an overlay on top of an interactive 2D map. The two strategies actually complement one another, and serve different purposes. The former, even if enhanced (e.g., LineDrive [2], Detail Lenses [11]), is designed to match drivers' cognitive representation of a route they are following in real-time [15]. However, they fail, because of their linear nature, to support the initial exploration and planning phase. The latter strategy better supports this phase, letting users freely explore the area surrounding the points of interest that the itinerary will go through, and plan alternative routes.

But interactive maps usually require users to perform a lot of panning and zooming actions to both see the complete itinerary and look at particular details, possibly causing some disorientation. One option to address this issue consists in displaying portions of the itinerary at different scales, for instance using PolyZoom [10], a visualization that organizes multiple linked views of the same map at different scales in a hierarchical manner, letting users pan & zoom any of those. The technique has been shown to outperform pan & zoom for some multi-scale search tasks, but can be cognitively demanding as it requires users to manage and mentally relate numerous spatially-disconnected views.

Focus+context techniques address this latter issue by smoothly integrating a zoomed-in representation of the current area of interest into a smaller-scale overview of the entire map [4]. The wired fisheye lens [17] lets mobile users explore their surroundings on a map by tilting their device. But the technique ties the lens to a particular point on the map using a rubber-wire-like mechanism, and does not help steer along paths [1]. JellyLenses [13] adapt their shape to the geometry of the objects they magnify, so as to optimize the visual representation of the focus, context, and transition areas. However, they are only concerned with optimizing the visual representation when performing magnified steering, and do not consider the motor aspects of the task. High-Precision Magnification Lenses [4] do look into the motor aspects of lens positioning, but do not provide support for steering.

While fisheyes have been shown to perform well for large steering tasks [8], those tasks remain challenging ones, in part because paths have a tendency to "slip off" the side of the lens. This specific problem could be addressed by combining lenses with LinkSliding [12], a technique for navigating large networks that strictly constrains movements to the path itself. However, as it translates the viewport (and thus the context view) whenever the cursor moves, it makes exploration of a route's surroundings impractical.

3. ROUTELENS

A *RouteLens* facilitates steering along a route by behaving as if it were attracted by it. This behavior is achieved by decoupling the lens' position from the cursor's position, following an approach similar to that of Semantic Pointing [6], that enlarges targets of interest in motor space by decoupling the latter from its visual counterpart. When using a *RouteLens*, all route segments whose distance to the system cursor is less than Δ apply an attraction force to the lens. The lens' position L is computed as a function of the system cursor's position C by using a weighted mean between all attracting route segments:

$$L = C + d_{min} \cdot \frac{\sum_{i=1}^{n} w_i \cdot A_i}{\sum_{i=1}^{n} w_i}$$

where A_i is the force vector that route segment i applies at position C to attract the lens (see below) and d_{min} is the distance between the cursor and the closest route segment. To ensure continuous lens movements when a route segment starts or stops having an influence on the lens, w_i is set to $\Delta - d_{c,i}$, where $d_{c,i}$ is the distance between the cursor and route segment i.

For a given route segment, the attraction vector is computed as:

$$A = \alpha(d_c) \cdot (R_c - C)/d_c$$

where R_c is the point on the route closest to the cursor, and d_c the distance between the cursor and the route segment. α is a power function of d_c that parameterizes the force vector a route segment applies to the lens:

$$\alpha(d_c) = \begin{cases} 1 - \left(\frac{d_c}{\Delta}\right)^p & \text{if } d_c \leq \Delta \\ 0 & \text{otherwise.} \end{cases}$$

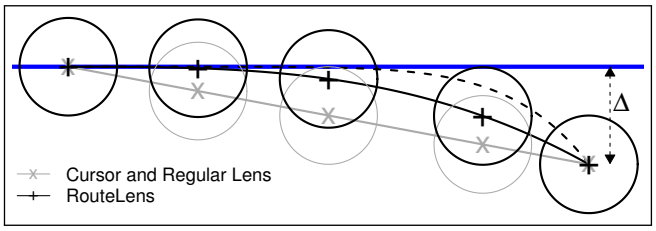

Figure 2: Position of the cursor (grey line), and of the RouteLens (black line), that is vertically attracted ($p = 2$) by the route (bold blue line). The dashed black line shows the positions of the RouteLens when $p = 6$. In this figure, Δ is equal to the lens' flat-top diameter in motor space, and the black (resp. grey) circles show the part of the context displayed in the RouteLens' (resp. regular fisheye lens') flat-top.

Figure 2 illustrates the progressive attraction effect of a straight route on the lens' position. As the cursor gets away from the route, the lens moves away from it more slowly than the cursor does. It then progressively moves faster, so as to match the cursor's position as soon as the latter leaves the attraction area. This area corresponds to a tunnel of diameter Δ centered on the route. Power parameter p is used to fine-tune the attraction effect, controlling how quickly the lens leaves the route to match the cursor's position. The solid and dashed black lines in Figure 2 illustrate two different p values.

Our composition of different attraction forces is in the spirit of techniques that rely on force vectors to adapt control-display gain [3, 7, 9]. Among these, Kinematic Templates [7] are the closest to our approach, as they consider fixed field vectors around an arbitrary path to create visual magnetic guides that can be combined together to generate artistic effects in a mouse-operated drawing application. Our approach is also related to that employed in Snap-and-go [5]. But while this technique works well for 1D line snapping, it is not straightforward to generalize to arbitrary 2D routes. It also makes users feel like the cursor temporarily stops and potentially causes exaggerated movements compared to the more progressive transitions that RouteLens enables.

When steering along a magnified route, users want to minimize the distance d_l between the lens' center and the route. In Accot

& Zhai's steering law [1], d_l represents the movement's variability along the tunnel centered on the route, i.e., the tunnel's width. The law stipulates that the larger the variability, the easier the movement. Figure 2 shows how *RouteLens* makes steering easier than a regular fisheye lens does, by allowing for a wider variability in user-controlled cursor movements. To keep a regular lens at a distance d_l from the route, users have to keep the cursor at a distance $d_c = d_l$. With a *RouteLens*, this distance can be larger: $d_c = d_l + d_c \cdot \alpha(d_c)$.

With a regular fisheye lens, keeping the route visible in the flat-top requires that variability d_l be less than $\frac{F}{M}$ (half of the tunnel width according to the steering law), where F is the radius of the flat-top, and M the magnification factor. With a *RouteLens*, the tunnel width can be larger if the radius of the attractive area goes beyond $\frac{F}{M}$ (the actual tunnel width is $2\sqrt[p+1]{\frac{F \cdot \Delta^p}{M}}$). Preliminary testing and pilot studies showed that setting $\Delta = \frac{2F}{M}$ and $p = 2$ yield a good balance between the magnitude of the attraction and the smoothness of the transition, resulting in an overall lens behavior that is hardly noticed by users. Figure 2 illustrates how these values make the maximal variability of the cursor movement ~ 1.6 larger than a regular fisheye lens does.

4. EVALUATION

RouteLenses' motor behavior can be implemented with any type of lenses, including the shape-shifting JellyLenses [13] that adapt their geometry to that of objects of interest or the high-precision magnification lenses [4] that address problems of quantization at high magnification factors. In this study we consider conventional fisheye lenses as a baseline to both isolate the benefits of RouteLenses' motor behavior and keep a reasonable experiment length for participants. The experimental task consisted in following a route with a lens, always keeping the route visible in the flat-top (Figure 3). We hypothesized that *RouteLens* outperforms *RegularLens* on this route following task, since the former facilitates the underlying steering task [1], as discussed above.

Participants

12 volunteers (6 female), all right-handed, aged 23 to 39 years-old (average 28.2, median 27), daily mouse users, participated in the experiment. 11 use mapping applications frequently. 7 of them are familiar with magnifying lenses, and 5 with fisheye lenses.

Apparatus

We conducted the experiment on a Mac Pro workstation running Mac OS X, equipped with an ATI Radeon HD 5870 video card driving a 30" LCD monitor (2560×1600, 100 dpi), and a standard optical mouse (800 dpi resolution) set with the default system acceleration. The software was implemented in JavaScript and WebGL using three.js (`http://threejs.org/`) and ran in Google Chrome. We used fisheye lenses with a magnification factor of 4, a lens size of 260 pixels and a flat top size of 180 pixels.

Design and Procedure

Our goal was to evaluate the motor aspects of the magnified steering task, focusing on the interaction technique itself. We thus decided not to use a real map to avoid any noise and bias due to the additional information that it would have featured. Instead, we operationalized the task using quantitative description factors. We could better control those, while still ensuring that the task was sufficiently representative of actual route following.

The experiment is a 2×2×2×4 within-subjects design with factors: TECH, ANGLE, DISTRACTOR, and DIR. TECH is the primary factor with two values: *RegularLens* and *RouteLens*. ANGLE and DISTRACTOR are secondary factors that define characteristics of the route, illustrated in Figure 3: ANGLE (*Acute* $= \pi/4$ and *Obtuse* $= 3\pi/4$) defines the angle between two route segments. DISTRACTOR defines the presence or absence of distractor routes, that also attract the cursor. When DISTRACTOR = *With*, additional (grey) routes are added at each turn of the (black) target route. DIR defines the direction of steering: left-to-right, right-to-left, top-to-bottom, or bottom-to-top. This factor was introduced for ecological reasons, and we do not consider it further in our analyses.

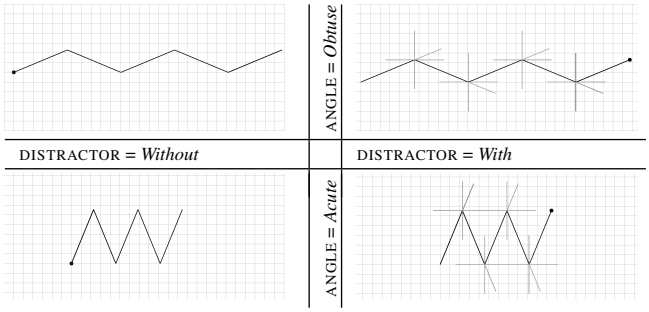

Figure 3: The four path configurations used in the experiment. The target route (black) always consists of 5 segments, 300 pixels in length each. Distractor routes are painted gray.

To start a trial, participants have to click on a black circle at the beginning of the first segment of the route. They then follow the target route and click on a rectangle located at the route's other endpoint. They are instructed to keep the route visible in the lens' flat-top all the way. As soon as the route leaves the flat-top, participants get notified on-screen and have to restart the whole steering task until they succeed. For each trial, we record (i) the (successful) task completion time *TCT*; (ii) the number of failed attempts, *NumError*; and (iii) the average distance from the lens to the target route.

We grouped trials into 2 blocks, one per technique, half of the participants starting with *RouteLens*. A block contains four sub-blocks of 16 trials corresponding to all ANGLE × DISTRACTOR × DIR conditions, presented in a random order. The first sub-block is for training purposes only. We collect measures for analyses during the three other sub-blocks. The overall experiment lasts about 50 minutes, including final debriefing and questionnaire.

4.1 Results

Figure 4 shows trial completion time *TCT* for each TECH by DISTRACTOR, and the results of the ANOVA for the full factorial model *TCT* $\sim$ TECH × DISTRACTOR × ANGLE × Rand(PARTICIPANT).

TECH has a significant effect on *TCT*: *RouteLens* is significantly faster than *RegularLens*, a difference of $\sim 15\%$. We also observe significant effects of DISTRACTOR and ANGLE on *TCT*: tasks are $\sim 5\%$ faster with distractors than without, and $\sim 6\%$ faster with *Acute* angles than with *Obtuse* angles.

More importantly, we observe a significant TECH × DISTRACTOR interaction (Figure 4-b). Post-hoc t-tests with Holm correction for multiple pair comparisons reveal that *RouteLens* is faster than *RegularLens*, both with distractors ($p < 0.0001$) and without ($p = 0.0002$). However, distractors have a stronger impact on *RouteLens*. *RouteLens* is significantly faster with distractors than without, but this DISTRACTOR effect is not significant for *RegularLens* ($p = 0.1420$). The presence of distractors actually improves *TCT* for *RouteLens* without introducing more errors ($p = 0.98$).

Effect for *TCT*	n, d	$F_{n,d}$	p
TECH	1,11	16.6	0.0018 **
DISTRACTOR	1,11	11.3	0.0062 **
ANGLE	1,11	6.45	0.0274 *
TECH × DISTRACTOR	1,11	7.92	0.0168 *
TECH × ANGLE	1,11	3.32	0.0955
DISTRACTOR × ANGLE	1,11	0.05	0.8287
TECH × DISTRACTOR × ANGLE	1,11	0.60	0.4540

(a)

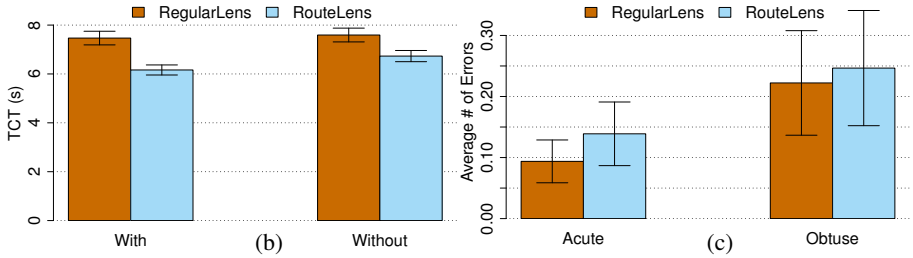

Figure 4: (a) ANOVA **for the full factorial model** *TCT* $\sim$ TECH $\times$ DISTRACTOR $\times$ ANGLE $\times$ $Rand$(PARTICIPANT)**. (b)** *TCT* **by** TECH×DISTRACTOR **condition. (c) Average number of errors by trial, by** TECH $\times$ ANGLE **condition. Error bars show the confidence interval for the mean over all trials.**

This may be due to the specific route layout we considered: the presence of a distractor route in the middle of the turn may make the turning movement easier. This middle route applies additional force vectors, resulting in a stronger global attraction towards the route at the end of the turn.

Regarding the number of errors *NumError*, we observe a significant difference for ANGLE only. Participants made more errors (Figure 4-c) with *Obtuse* than with *Acute* angles ($p = 0.0072$). Thus, participants were both slower (see above) and made more errors in the *Obtuse* condition. It may appear as a surprising result, as movements that involve highly curved portions are generally slower [16] than straighter movements. We attribute this to the more frequent mouse clutching actions the operator noticed for *Obtuse* angles, the path being lengthier in the direction of steering (see Figure 3).

As expected, *RouteLens*' attraction effect also makes users steer along the route with a movement that exhibits less variability. The average distance from the lens' center to the route is significantly lower for *RouteLens* than for *RegularLens* ($F_{1,11} = 331$, $p < 0.0001$). The distance is 13.3 ± 0.9 pixels for *RouteLens* and 33.3 ± 1.7 pixels for *RegularLens* (expressed with respect to the flat-top's coordinate system).

At the end of the experiment, we collected qualitative feedback using a post-hoc questionnaire. Three participants did not even notice the difference between the regular fisheye and RouteLens. Among the nine participants who did notice a difference, three participants said they had no idea what that difference was (they just felt like they could go faster with RouteLens) and seven had a preference for *RouteLens* (only one had a preference for *RegularLens*).

5. CONCLUSION

RouteLenses are designed to make it easier for users to follow map itineraries by dynamically adapting properties of the motor space, based on both cursor position and route geometry. This is achieved without constraining users, who remain free to explore the itinerary's surroundings without having to perform any explicit action to either engage or disengage the modified motor behavior. In our laboratory experiment, RouteLens better helped users stay close to the path they follow than a regular fisheye did, without making the lens stick too strongly to it.

Future work will evaluate RouteLens' performance beyond steering, e.g., when disengaging from the route to explore its local surroundings. We would also like to further study the effect of the number and geometrical configuration of distractors routes present at intersections. We also plan to investigate the combination of RouteLenses' motor behavior with techniques designed for high magnification factors [4] and dynamic visual adaptation [13].

6. REFERENCES

[1] J. Accot and S. Zhai. Beyond Fitts' law: models for trajectory-based HCI tasks. *CHI '97*, 295–302. ACM, 1997.

[2] M. Agrawala and C. Stolte. Rendering effective route maps: improving usability through generalization. *SIGGRAPH '01*, 241–249. ACM, 2001.

[3] D. Ahlström, M. Hitz, and G. Leitner. An evaluation of sticky and force enhanced targets in multi target situations. *NordiCHI '06*, 58–67. ACM, 2006.

[4] C. Appert, O. Chapuis, and E. Pietriga. High-precision magnification lenses. *CHI '10*, 273–282. ACM, 2010.

[5] P. Baudisch, E. Cutrell, K. Hinckley, and A. Eversole. Snap-and-go: Helping users align objects without the modality of traditional snapping. *CHI '05*, 301–310. ACM, 2005.

[6] R. Blanch, Y. Guiard, and M. Beaudouin-Lafon. Semantic pointing: improving target acquisition with control-display ratio adaptation. *CHI '04*, 519–526. ACM, 2004.

[7] R. Fung, E. Lank, M. Terry, and C. Latulipe. Kinematic templates: end-user tools for content-relative cursor manipulations. *UIST '08*, 47–56. ACM, 2008.

[8] C. Gutwin and A. Skopik. Fisheyes are good for large steering tasks. *CHI '03*, 201–208. ACM, 2003.

[9] A. Hurst, J. Mankoff, A. K. Dey, and S. E. Hudson. Dirty desktops: Using a patina of magnetic mouse dust to make common interactor targets easier to select. *UIST '07*, 183–186. ACM, 2007.

[10] W. Javed, S. Ghani, and N. Elmqvist. PolyZoom: multiscale and multifocus exploration in 2D visual spaces. *CHI '12*, 287–296. ACM, 2012.

[11] P. Karnick, D. Cline, S. Jeschke, A. Razdan, and P. Wonka. Route visualization using detail lenses. *IEEE TVCG*, 16(2):235–247, Mar. 2010.

[12] T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, and J.-D. Fekete. Topology-aware navigation in large networks. *CHI '09*, 2319–2328. ACM, 2009.

[13] C. Pindat, E. Pietriga, O. Chapuis, and C. Puech. JellyLens: content-aware adaptive lenses. *UIST '12*, 261–270. ACM, 2012.

[14] M. Sarkar and M. H. Brown. Graphical fisheye views of graphs. *CHI '92*, 83–91. ACM, 1992.

[15] B. Tversky. Cognitive maps, cognitive collages, and spatial mental models. *COSIT '93*, 14–24. Springer, 1993.

[16] P. Viviani and C. Terzuolo. Trajectory determines movement dynamics. *Neuroscience*, 7(2):431–437, 1982.

[17] D. Yamamoto, S. Ozeki, and N. Takahashi. Wired fisheye lens: A motion-based improved fisheye interface for mobile web map services. *W2GIS '09*, 153–170. Springer, 2009.